

Search Login Register Order Form Shopping Cart Premium Features



JP5257666A2: AUTOMATIC FLOWCHART GENERATING METHOD

[View Images \(1 pages\)](#) | [View INPADOC only](#)

Country

JP Japan

Kind

Inventor(s)

MATSUMOTO SHUICHI

Applicant(s)

HITACHI LTD
HITACHI SOFTWARE ENG CO LTD
[News, Profiles, Stocks and More about this company](#)

Issued/Filed Dates

Oct. 8, 1993 / March 13, 1992

Application Number

JP1992000055084

IPC Class

G06F 9/06;

Abstract

Purpose: To automatically generate and output a flowchart corresponding to an analytic viewpoint by varying a variable which becomes a flowchart output condition in accordance with a desired analytic viewpoint.

Constitution: In a source code 1 of a computer program, a display condition setting sentence 10 and a display condition deciding sentence 11 set by a user are embedded. A display range determining means 2 analyzes the source code 1, and determines a range of the source code which becomes a generating object of a flowchart, based on the display condition setting sentence 10 and the display condition deciding sentence 11 embedded in the source code 1, and display conditions given from a display condition setting means 6 and a display condition calculating means 7. A flowchart display means 3 generates a flowchart 5, based on the determined display range. At time of generation, a value of a set variable is decided, and the flowchart 5 is generated by excluding a program part on a syntax in which a deciding sentence having a variable which does not correspond to the set variable exists.

COPYRIGHT: (C)1993,JPO&Japio

Family

[Show known family members](#)

Other Abstract Info:

none

Foreign References:

No patents reference this one

Patent Plaques

Recognize the achievement



[Nominate this invention](#)

Alternative Searches

[Patent Number](#)

[Boolean Text](#)

[Advanced Text](#)

Browse

[U.S. Class by title](#)

[U.S. Class by number](#)

[IP Listing Search](#)

BEST AVAILABLE COPY

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平5-257666

(43) 公開日 平成5年(1993)10月8日

(51) Int.Cl.⁵

G 0 6 F 9/06

識別記号

4 3 0 H 8944-5B

庁内整理番号

F I

技術表示箇所

審査請求 未請求 請求項の数4 (全 14 頁)

(21) 出願番号 特願平4-55084

(22) 出願日 平成4年(1992)3月13日

(71) 出願人 000005108

株式会社日立製作所

東京都千代田区神田駿河台四丁目6番地

(71) 出願人 000233055

日立ソフトウェアエンジニアリング株式会
社

神奈川県横浜市中区尾上町6丁目81番地

(72) 発明者 松本 秀一

神奈川県横浜市中区尾上町6丁目81番地

日立ソフトウェアエンジニアリング株式会
社内

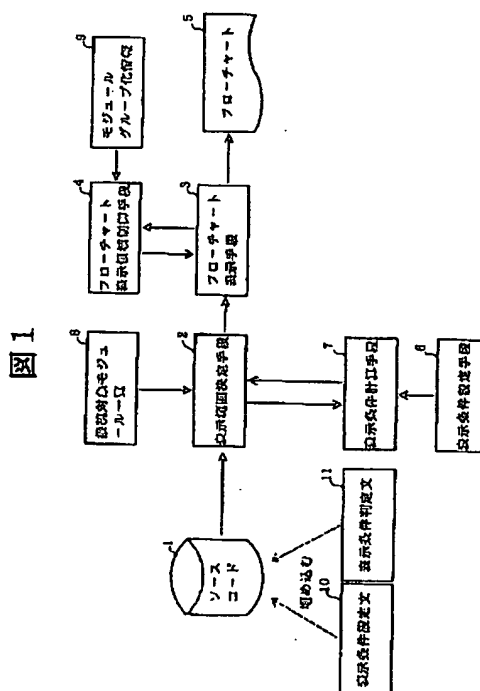
(74) 代理人 弁理士 秋田 収喜

(54) 【発明の名称】 フローチャート自動生成方法

(57) 【要約】

【目的】 複雑なコンピュータプログラムの流れを特定の
実行条件や観点に基づいて容易に把握できるフローチャ
ートを自動生成する。

【構成】 フローチャートの出力範囲を制御する判定文
をコンピュータプログラム中に埋め込んでおき、フロー
チャートの生成に際して設定した変数の値を判定文により
判定し、設定した変数に対応しない変数を持つ判定文
が存在する構文上のプログラム部分についてのみフロー
チャートを生成し、出力する。



【特許請求の範囲】

【請求項1】 コンピュータプログラムのソースコードを解析してフローチャートを自動的に生成して出力する方法であって、フローチャートの出力範囲を制御するための判定文を解析対象のコンピュータプログラムに予め埋め込んでおき、フローチャートの生成に際して設定した変数の値を判定文により判定し、設定した変数に対応しない変数を持つ判定文が存在する構文上のプログラム部分を除外してフローチャートを生成し、出力することを特徴とするフローチャート自動生成方法。

【請求項2】 コンピュータプログラムのソースコードを解析してフローチャートを自動的に生成して出力する方法であって、フローチャートの出力範囲を制御するための判定文を解析対象のプログラムに予め埋め込んでおき、さらに他のプログラムモジュールがコールされる位置には該他のプログラムモジュールの実効条件となる変数を設定する設定文を予め埋め込んでおき、フローチャートの生成に際して設定した変数の値を判定文により判定し、設定した変数に対応しない変数を持つ判定文が存在する構文上のプログラム部分を除外したプログラム部分と、該プログラム部分に関連して実行されるプログラムモジュールのうち前記設定文によって実行条件となる変数が設定されたプログラムモジュールの処理の流れとを接続し、一つのフローチャートとして生成し、出力することを特徴とするフローチャート自動生成方法。

【請求項3】 前記プログラムモジュールの処理の流れを示す部分は、プログラムモジュールのグループ別に区分けして出力することを特徴とする請求項2記載のフローチャート自動生成方法。

【請求項4】 ソースコード中に、フローチャートに表示する旨の印を付けた注釈をさらに埋め込んでおき、その注釈と特定の事象を引き起こすコードが存在する部分のプログラムの骨格構造だけを概略のフローチャートとして出力することを特徴とする請求項1または2記載のフローチャート自動生成方法。

【発明の詳細な説明】

【0001】

【産業上の利用分野】 本発明は、コンピュータプログラムのソースコードを解析してフローチャートを自動的に生成して出力するフローチャート自動生成方法に関する。

【0002】

【従来の技術】 従来において、コンピュータプログラムのソースコードからフローチャートを自動的に生成する方法として、例えば、「日立評論、Vol. 70, No. 2, p7~14 (昭63-2)」に紹介されたSEWB (ソフトウェアエンジニアリングワークベンチ) が知られている。これは、ソースコードをフローチャートの一種であるPAD図 (Problem Analysis Diagram) に変換するものである。

【0003】 また、特開平2-39324号公報に開示されているように、処理要素ごとに、フローチャートへの表示出力の有無および表示出力する場合の付加情報を定義することにより、概略フローチャートを生成する方法がある。

【0004】 さらに、特開平2-48731号公報に開示されているように、プログラムの構造要素ごとに、概略レベルを定義しておき、フローチャート生成時に、生成したいフローチャートの概略レベルを指定することにより、そのレベルより詳細な部分を削除して概略フローチャートを生成する方法がある。

【0005】

【発明が解決しようとする課題】 しかしながら、SEWB (ソフトウェアエンジニアリングワークベンチ) は、ソースコードの文と1対1にPAD図の処理ボックスを生成するため、様々な条件の処理が混在した複雑なプログラムでは、生成されるフローチャートが複雑なものとなり過ぎ、広い範囲のプログラムの流れを把握したり、解析する目的には適さないという問題があった。

【0006】 また、特開平2-39324号および特開平2-48731号のフローチャート生成方法で生成されるフローチャートは、特定の実行条件や観点に基づいた概略フローチャートではないので、様々な条件の処理が混在した複雑なプログラムの流れを把握したり解析するには、不十分であるという問題があった。

【0007】 本発明の目的は、様々な条件で実行される処理が混在した複雑な流れを持つコンピュータプログラムについて、その流れを特定の実行条件や観点に基づいて容易に把握することが可能な広い範囲にわたる見通しの良いフローチャートを自動生成することができるフローチャート自動生成方法を提供することである。

【0008】

【課題を解決するための手段】 上記目的を達成するために本発明は、基本的には、フローチャートの出力範囲を制御するための判定文を解析対象のコンピュータプログラムに予め埋め込んでおき、フローチャートの生成に際して設定した変数の値を判定し、設定した変数に対応しない変数を持つ判定文が存在する構文上のプログラム部分を除外してフローチャートを生成し、出力するようにした。

【0009】 あるいは、フローチャートの出力範囲を制御するための判定文を解析対象のプログラムに予め埋め込んでおき、さらに他のプログラムモジュールがコールされる位置には該他のプログラムモジュールの実効条件となる変数を設定する設定文を予め埋め込んでおき、フローチャートの生成に際して設定した変数の値を判定し、設定した変数に対応しない変数を持つ判定文が存在する構文上のプログラム部分を除外したプログラム部分と、該プログラム部分に関連して実行されるプログラムモジュールのうち前記設定文によって実行条件となる変

数が設定されたプログラムモジュールの処理の流れとを接続し、一つのフローチャートとして生成し、出力するようにした。

【0010】

【作用】上記手段によれば、所望の解析観点のプログラム部分の流れだけを示すフローチャートを出力しようとする場合、フローチャートの生成に先だって、ソースコード中にフローチャートとしての出力条件を記述した判定文を予め埋め込んでおく。そして、フローチャート生成時には、フローチャート出力条件となる変数を設定する。すると、この設定した変数に対応しない変数を持つ判定文が存在する構文上のプログラム部分を除外したプログラム部分についてのみのフローチャートが生成されて出力される。

【0011】従って、フローチャート出力条件となる変数を所望の解析観点に応じて変えることにより、その解析観点に応じたフローチャートを自動的に生成して出力することができる。これにより、様々な条件で実行される処理が混在した複雑な流れを持つコンピュータプログラムであっても、その流れを特定の実行条件や観点に基づいて容易に把握することが可能になる。

【0012】また、他のプログラムモジュールを呼び出して処理を進めている部分では、設定文によって実効条件となる変数が設定されたプログラムモジュールの処理の流れが接続され、1つのフローチャートとして出力される。

【0013】

【実施例】以下、図示する実施例に基づいて本発明を詳細に説明する。

【0014】図1は本発明を適用したフローチャート自動生成システムの一実施例の構成図を示すものである。図において、コンピュータプログラムのソースコード1の中には、ユーザが設定した表示条件設定文10と表示条件判定文11が埋め込まれている。

【0015】表示範囲決定手段2はソースコード1を解析し、該ソースコード1中に埋め込まれている表示条件設定文10と表示条件判定文11と、表示条件設定手段6及び表示条件計算手段7から与えられる表示条件に基づき、フローチャートの生成対象となるソースコードの範囲を決定する。このとき、表示範囲決定手段2は、複数のプログラムモジュールのうち接続して実行するプログラムモジュールの接続対象モジュール一覧8が指定されていれば、それらのモジュールの全てについて表示範囲の決定を行う。

【0016】フローチャート表示手段3は、決定された表示範囲をもとに、フローチャート5を生成する。このとき、フローチャート表示手段3は、モジュールグループ化情報9が指定されていれば、フローチャート表示領域制御手段4からの制御に従ってフローチャートの表示領域を各モジュールグループに対応する領域に分割して

表示する。

【0017】図2に表示条件設定文10の記述例を示す。この表示条件設定文10は、「/*!」で始まる注釈であり、フローチャートの表示範囲を制御する変数、例えば「COUNT」に対する代入文を含む。この設定文10は、1つのプログラムモジュールに関連して実行されるプログラムモジュールの実行条件となる変数の値を設定する場合に使用される。

【0018】図3に表示条件判定文11の記述例を示す。この表示条件判定文11は、「/*?」で始まる注釈であり、フローチャートの表示範囲を制御する変数、例えば「MODE及びCOUNT」に関する論理式を含む。

【0019】図4は表示条件設定手段6で表示条件として設定する変数の指定例を示すもので、フローチャート生成時に、フローチャートの表示範囲を制御する変数（この例では、MODE）に対する代入文として、UTLを指定した例を示している。

【0020】この実施例においては、表示条件の変数として、MODE=UTLを設定した場合、MODE=UAPという内容を持つ判定文が埋め込まれたプログラムのブロックがフローチャートの生成対象から除外される。

【0021】図5は、PL/Iで記述したソースコードの例を示すもので、この例のソースコードに対し、図4で示した表示条件MODE=UTLを表示条件設定手段6から指定すると、図6に示すようなフローチャートの一種であるPAD図が生成される。

【0022】すなわち、図5の例のソースコードにおいては、

- (1) MODE=UTL
- (2) MODE=UAP
- (3) FUNC=ADD
- (4) FUNC=DEL

という変数を持つ判定文50、51、52、53が記述されているが、表示条件設定手段6では、MODE=UTLが設定されたので、判定文50が記述されている部分の「BUF=30; /*パッファ面数*/」がフローチャート生成対象として選択され、図6に符号60で示すようなフローチャートとして生成される。

【0023】次に、「IF NUM>0 THEN」の条件実行文には、

- (1) FUNC=DEL
- (2) FUNC=ADD

という設定文54、55が記述され、判定文52、53を持つプログラムモジュールMOD1を呼び出すための条件が設定される。プログラムモジュールMOD1においては、FUNC=DELに設定されていた場合は判定文52により「NUM=NUM+1」が表示対象から除外され、FUNC=ADDに設定されていた場合は判定

文53により「NUM=NUM-1」が表示対象から除外されるようになっているので、この部分は符号61で示すようなフローチャートとして生成される。

【0024】一方、「IF RC=ERROR THEN」の条件実行文においては、判定文が記述されていないので、そのまま符号62で示すフローチャートとして生成される。

【0025】この場合、図5の例では、プログラムモジュールのグループ化をしていないが、モジュールのグループ化情報9を指定した場合には、図7の例に示すように、各プログラムモジュールのグループに対応してフローチャート表示領域が横方向に分割され、グループ内のフローは対応する表示料領域の中にそれぞれ区分けして表示される。なお、図7で示す各処理の内容は図5のソースコードとは何等関係ない。

【0026】このようにグループ分けしてフローチャートを表示出力することにより、各プログラムモジュールの接続関係を明瞭に把握することが可能になる。

【0027】図8に、表示条件計算手段7による表示条件の計算例を示すものである。表示条件計算手段7は、20 ソースコード1の構文を解析して、図8に示す構文上のパターンを認識し、そのパターンの直後の場所の表示条件を図8に示すような計算方法で計算する。

【0028】すなわち、ソースコードの解析を進めて行く過程で分岐判定部80があり、その分岐先として「V=A」、「V=B」の設定文81、82が設定されていた場合は、「V=A」、「V=B」という計算結果を得、これを以降の表示条件の変数として用いる。

【0029】また、ソースコードの解析を進めて行く過程でループ部83があり、その先に「V=A」の設定文30 84が設定されていた場合は、「V=A」、「V=φ（空欄）」という計算結果を得、これを以降の表示条件の変数として用いる。

【0030】また、ソースコードの解析を進めて行く過程で「V=A」、「V=B」の設定文85、86が設定されていた場合は、最終的に「V=B」という計算結果を得、これを以降の表示条件の変数として用いる。

【0031】なお、この計算は、ソースコードのプログラムを実際に実行するわけではなく、静的な構文解析を行うだけなので、実用的な時間内に終了する。この計算結果は、プログラムモジュールの実行条件となる変数40 がとりえる値の候補であり、実際にプログラムを実行したときにはありえない値を含む可能性があってフローチャートの表示範囲が多少広がるかもしれないが、プログラムの流れの前後関係によって決定される表示条件を求めるのには都合が良く、ソースコードをエンハンスによって修正しても表示条件設定文や表示条件判定文の修正が不要となる場合が多いので、実際上は極めて都合がよい。

【0032】次に、図1のフローチャート自動生成シス 50

テムにおけるフローチャート生成出力手順を図9および図10のPAD図を用いて説明する。

【0033】最初に、表示条件設定手段6により、フローチャートの表示範囲を制御する変数に初期値を設定する（ステップ901）。次に、フローチャート表示範囲決定手段2により、フローチャートを生成して表示するソースコード1の範囲を決定する（ステップ902）。その結果をもとに、フローチャート表示手段3がフローチャートの一種であるPAD図を生成する（ステップ903）。このとき、モジュールのグループ化情報9が与えられていれば、フローチャート表示手段3は、フローチャート領域制御手段4を用いて、フローチャート表示領域を各モジュールグループに対応する部分に分割し、図7の例に示したように表示する。

【0034】フローチャートの表示範囲決定手段2においては、まずソースコード1中の表示条件設定文10と、表示条件計算手段7が計算したソースコード1の各部におけるフローチャートの表示範囲を制御する変数の値をもとに、表示範囲を決定する（ステップ904）。

【0035】次に、現在解析中のモジュールのソースコードの中の全ての表示条件判定文11について、次の手順を繰り返す（ステップ905）。

【0036】（1）表示範囲決定手段2によって表示条件判定文11の真偽を判定する（ステップ906）。真偽を判定するとは、その判定文の中に設定手段6で設定された変数に一致する、あるいは対応する変数を持っているかどうかを判定することである。

【0037】（2）判定結果が偽であれば（ステップ907）、表示範囲決定手段2は、その判定文11の存在する構文上のブロックにおける判定文以降の部分、フローチャートの表示対象から除外する（ステップ908）。

【0038】次に、接続対象モジュール一覧8で指定されたモジュールに対する全てのコール文について、そのモジュールのソースコードについて表示範囲決定手段2を再起的に実行することを繰り返す（ステップ909、910）。

【0039】このように本実施例においては、フローチャートの出力範囲を制御するための判定文を解析対象のコンピュータプログラムに予め埋め込んでおき、フローチャートの生成に際して設定した変数に対応する変数を持つ判定文が存在する構文上のプログラム部分についてのみフローチャートを生成し、出力するようにしたため、フローチャート出力条件となる変数を所望の解析観点に応じて変えることにより、その解析観点に応じたフローチャートを自動的に生成して出力することができる。これにより、様々な条件で実行される処理が混在した複雑な流れを持つコンピュータプログラムであっても、その流れを特定の実行条件や観点に基づいて容易に把握することが可能になる。

【0040】また、関連して実行されるプログラムモジュールの処理の流れを接続し、一つのフローチャートとして生成し、出力しているため、複数のプログラムがどのような関連で動作するかを明瞭に把握することができるようになる。

【0041】図11はこの発明の第2の実施例を示す構成図であり、ソースコード中に埋め込んだ「フローチャートに表示する旨」の印を付けた注釈1012をもとにプログラムの枝葉末節な部分を省略したフローチャートを自動生成するようにしたものである。なお、図11と同一部分は同一記号で示している。

【0042】図において、コンピュータプログラムのソースコード1001には、表示条件設定文1010、表示条件判定文1011の他に、フローチャートに表示する旨の印を付けた注釈1012が埋め込まれている。

【0043】表示対象のモジュールコール文指定手段1013及び表示対象のマクロコール文指定手段1014は、それぞれ、概略フローチャートへの表示対象とするモジュールコール文のモジュール名と概略フローチャートへの表示対象とするマクロコール文のマクロ名を指定するものである。

【0044】図12は、フローチャートに表示する旨の印を付けた注釈1012の例を示す。この注釈は、「/*※」で始まる注釈であり、この注釈の内容が、そのまま概略フローチャートに表示される。

【0045】この図11の実施例のフローチャート自動生成システムは、ソースコード中の全ての実行文をフローチャートに表示せず、表示対象のモジュールコール文、表示対象のマクロコール文及び、表示する旨の印を付けた注釈だけをプログラムの骨格構造の中に埋め込んでフローチャートに表示するとともに、上記表示対象の項目を含まない、構文上のブロックは、プログラムの骨格構造から除外して表示するものである。

【0046】この実施例のフローチャート自動生成システムにおいては、PL/Iのソースコード1が図13に示すような内容であった場合、図14のようなPAD図が生成されて表示出力される。

【0047】図15および図16はそのフローチャート自動生成手順を示すPAD図である。これらのPAD図は図9、図10のPAD図とほぼ同じで、一部だけが異なる。従って、異なる部分のみを以下説明すると、図15、図16のPAD図の中のフローチャートの表示範囲決定処理(ステップ1402)においては、図16のステップ1411、1412に詳細を示しているように、表示対象の項目(表示対象のモジュールコール文、表示対象のマクロコール文及び、表示する旨の印を付けた注釈)を内部に含まない、構文上のブロックについて、そのブロックをフローチャートへの表示対象から除外する。この場合、ブロックがネストしており、内側のブロックに表示項目を含むなら、外側のブロックも表示項目

を含むと考える。また、フローチャートの表示処理(ステップ1403)においては、ソースコードの表示範囲の部分について、プログラムの骨格構造を表す結線の中に、上記表示対象の項目を埋め込んで表示するとともに、プログラムの表示対象のブロックについて、繰返しの条件や、分岐の条件をソースコードに記述してある通りに、フローチャートに表示する。

【0048】従って、図13の例のソースコード1にあっては、表示する旨の印を付けた注釈1012として、「バッファ面数の計算」という注釈1012A、「カウンタアップ」という注釈1012B、「カウンタダウン」という注釈1012Cがあり、符号141、142で示すように表示される。

【0049】このように、フローチャートに表示する旨の印を付けた注釈を埋め込み、さらに特定の事象を引き起こすコード(例えば、CALL)を認識し、それが存在する部分のプログラムの骨格構造だけを抽出し、そのフローチャートを生成するようにした場合、大規模なソフトウェアの大きな制御の流れを把握することが極めて容易になる。

【0050】

【発明の効果】以上のように本発明によれば、フローチャートの出力範囲を制御するための判定文を解析対象のコンピュータプログラムに予め埋め込んでおき、フローチャートの生成に際して設定した変数の値を判定文により判定し、設定した変数に対応しない変数を持つ判定文が存在する構文上のプログラム部分を除外してフローチャートを生成し、出力するようにしたため、フローチャート出力条件となる変数を所望の解析観点に応じて変えることにより、その解析観点に応じたフローチャートを自動的に生成して出力することができる。これにより、様々な条件で実行される処理が混在した複雑な流れを持つコンピュータプログラムであっても、その流れを特定の実行条件や観点に基づいて容易に把握することが可能になる。例えば、複雑なソフトウェアの保守において、変更に対する影響範囲を容易に把握できるフローチャートを作成し、解析を進めることできる。

【0051】また、フローチャートの出力範囲を制御するための判定文を解析対象のプログラムに予め埋め込んでおき、さらに他のプログラムモジュールがコールされる位置には該他のプログラムモジュールの実効条件となる変数を設定する設定文を予め埋め込んでおき、フローチャートの生成に際して設定した変数の値を判定文により判定し、設定した変数に対応しない変数を持つ判定文が存在する構文上のプログラム部分を除いたプログラム部分と、該プログラム部分に関連して実行されるプログラムモジュールのうち前記設定文によって実行条件となる変数が設定されたプログラムモジュールの処理の流れとを接続し、一つのフローチャートとして生成し、出力するようにした場合、複数のプログラムがどのような関

連で動作するかを明瞭に把握することができるようになる。

【0052】さらに、フローチャートに表示する旨の印を付けた注釈を埋め込み、さらに特定の事象を引き起こすコード（例えば、CALL）を認識し、それが存在する部分のプログラムの骨格構造だけを抽出し、そのフローチャートを生成するようにした場合、大規模なソフトウェアの大きな制御の流れを把握することが極めて容易になるという効果がある。

【図面の簡単な説明】

【図1】本発明を適用したフローチャート自動生成システムの一実施例を示す構成図である。

【図2】表示条件設定文の一例を示す説明図である。

【図3】表示条件判定文の一例を示す説明図である。

【図4】表示条件設定手段の表示条件の指定例を示す説明図である。

【図5】PL/Iのソースコードの一例を示す説明図である。

【図6】図5のソースコードに対応して生成出力されるPAD図である。

【図7】モジュールをグループ化した場合に生成出力されるPAD図である。

【図8】表示条件計算手段による計算例の一例を示す説明図である。

【図9】図1のシステムにおけるフローチャート生成手

順を示すPAD図である。

【図10】図9におけるフローチャート表示範囲決定処理の手順を示すPAD図である。

【図11】本発明を適用したフローチャート自動生成システムの他の実施例の構成図である。

【図12】フローチャートに表示する旨の印を付けた注釈の一例を示す説明図である。

【図13】PL/Iのソースコードの一例を示す説明図である。

10 【図14】図13のソースコードに対応して生成出力される概略PAD図である。

【図15】図10のシステムにおけるフローチャート生成手順を示すPAD図である。

【図16】図15におけるフローチャート表示範囲決定処理の手順を示すPAD図である。

【符号の説明】

1…コンピュータプログラムのソースコード、2…表示範囲決定手段、3…フローチャート表示手段、4…フローチャート表示領域制御手段、5…フローチャート、6…表示条件設定手段、7…表示条件計算手段、8…接続対象モジュール一覧、9…モジュールグループ化情報、10…表示条件設定文、11…表示条件判定文、101…フローチャートに表示する旨の印を付けた注釈、1013…表示対象モジュールコール文指定手段、1014…表示対象マクロコール文指定手段。

【図2】

図2

表示条件設定文の例

```
/*! COUNT=0; */
```

【図3】

図3

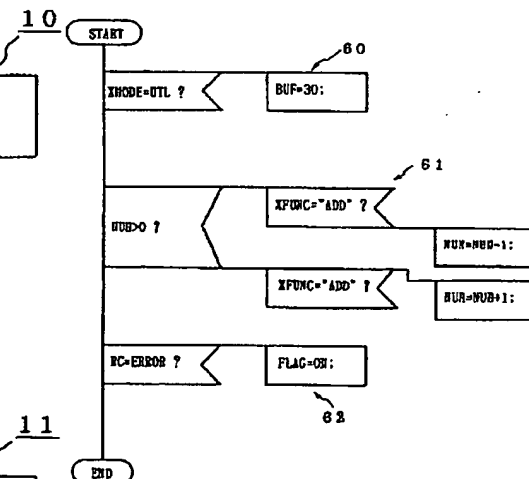
表示条件判定文の例

```
/*? (NODE="UTL" OR NODE="UAP")AND COUNT>0 */
```

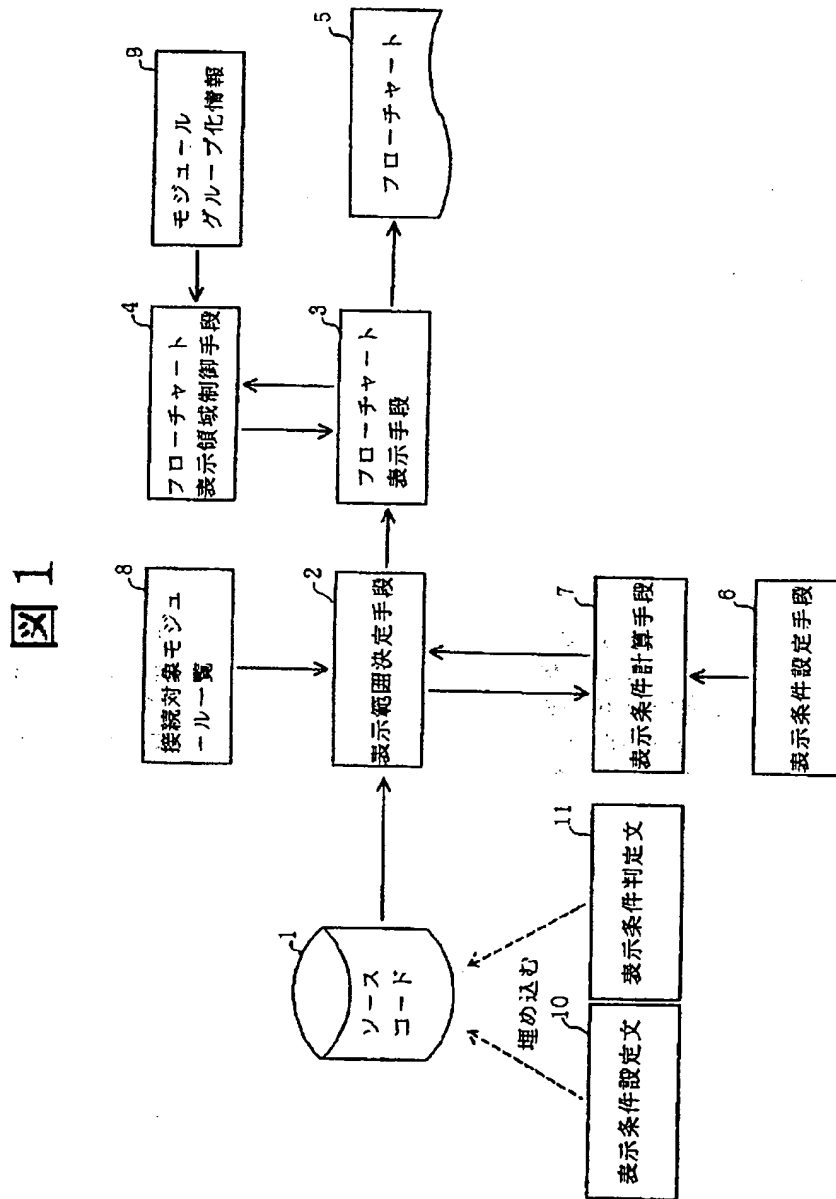
【図6】

図6

出力されるPAD図の例



【図1】



【図4】

図4

表示条件設定手段に対する指定例

```
MODE="UTL";
```

【図5】

図5

PL/Iのソースコードの例

```

PROCEDURE MAIN;
  IF XMODE=UTL THEN
    DO;
      /*? MODE="UTL" */ 50
      BUF=30; /* バッファ面数 */
    END;
  ELSE
    DO;
      /*? MODE="UAP" */ 51
      BUF=20; /* バッファ面数 */
    END;
  IF NUM>0 THEN
    DO;
      /*! FUNC="DEL"; */ 54
      CALL MOD1("DEL");
    END;
  ELSE
    DO;
      /*! FUNC="ADD"; */ 55
      CALL MOD1("ADD");
    END;
  /* エラ-処理 */
  IF RC=ERROR THEN
    FLAG=ON;
  ELSE;
END;

PROCEDURE MOD1(XFUNC);
  IF XFUNC="ADD" THEN
    DO;
      52 /*? FUNC="ADD" */
      NUM=NUM+1;
    END;
  ELSE
    DO;
      53 /*? FUNC="DEL" */
      NUM=NUM-1;
    END;
  END;

```

【図12】

図12

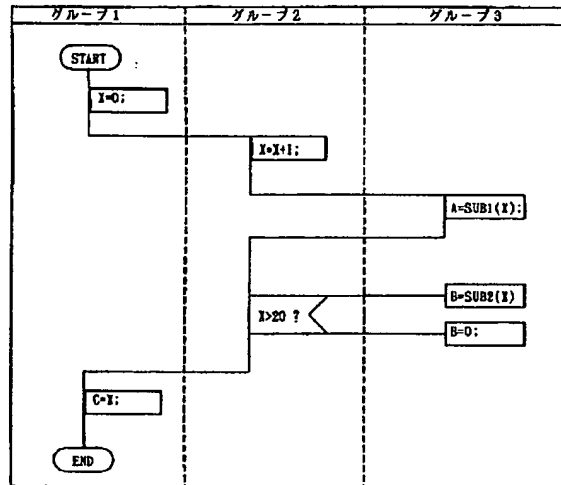
フローチャートに表示する旨の印を付けた注釈の例

```
/*? バッファ面数の計算 */
```

【図7】

図7

モジュールをグループ化した場合のPAD図の例

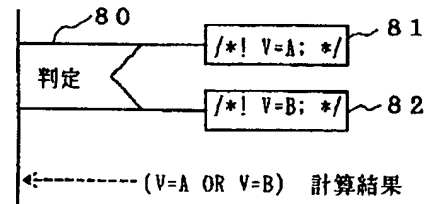


【図8】

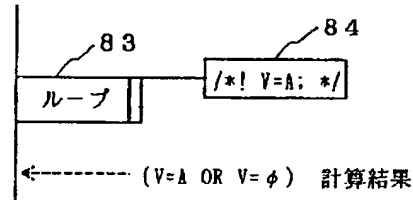
図8

表示条件計算手段による計算例

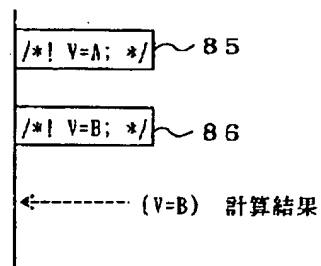
(a)



(b)

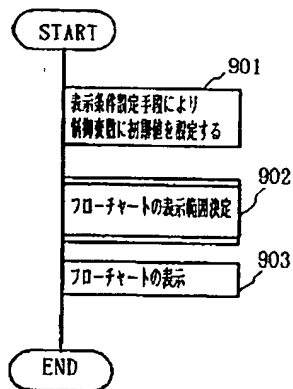


(c)



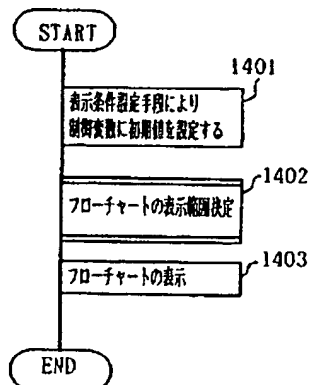
【図9】

図9



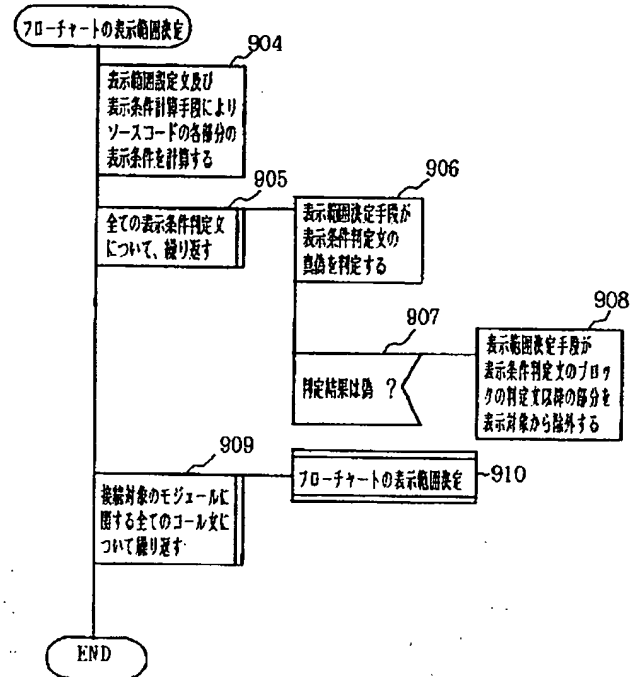
【図15】

図15



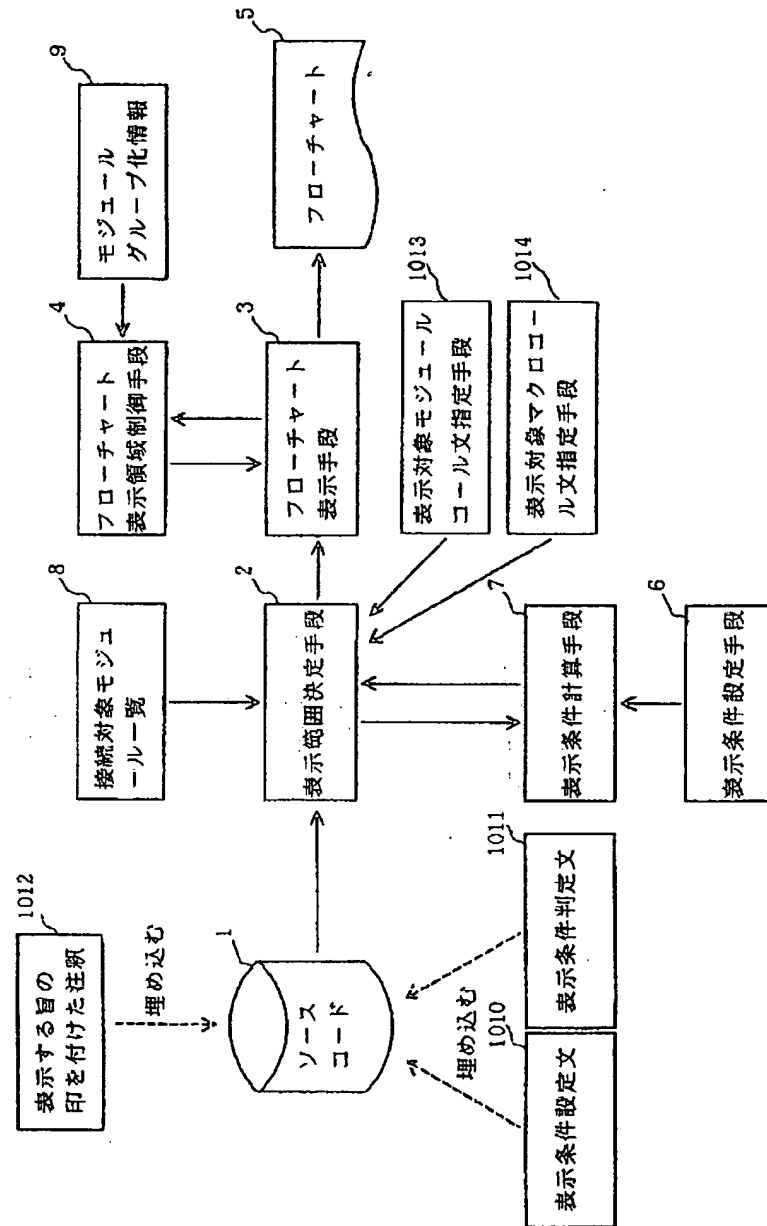
【図10】

図 10



【図11】

図11



【図13】

図 13

P L / I のソースコードの例

```

PROCEDURE MAIN;
  /*? パツパツ面数の計算 */
  IF XMODE=UTL THEN 1012A
    DO;
      /*? MODE="UTL" */
      BUF=30; /* パツパツ面数 */
    END;
  ELSE
    DO;
      /*? MODE="UAP" */
      BUF=20; /* パツパツ面数 */
    END;
  IF NUM>0 THEN
    DO;
      /*! FUNC="DEL"; */
      CALL MOD1("DEL");
    END; 1300
  ELSE
    DO;
      /*! FUNC="ADD"; */
      CALL MOD1("ADD");
    END; 1301
  /* エラ-処理 */
  IF RC=ERROR THEN
    FLAG=ON;
  ELSE;
END;

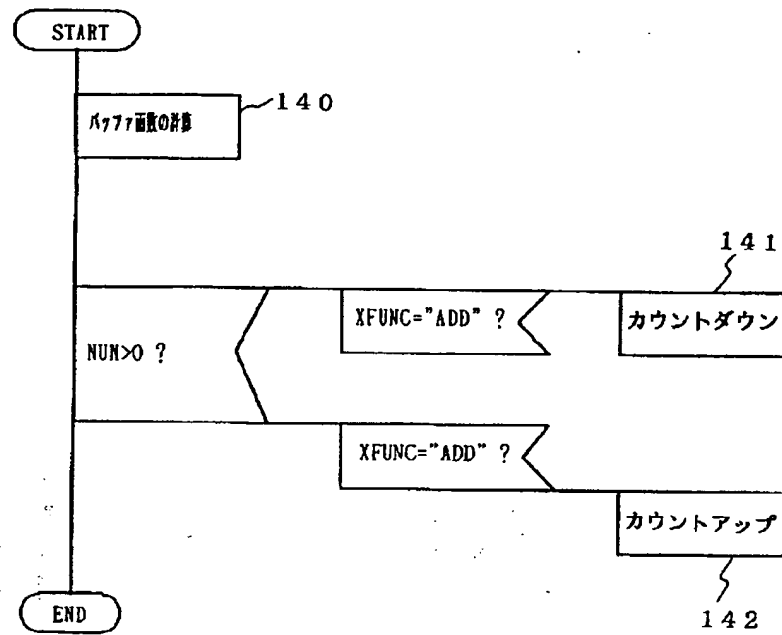
PROCEDURE MOD1(XFUNC);
  IF XFUNC="ADD" THEN
    DO;
      /*? FUNC="ADD" */
      /*? カウントアップ */
      NUM=NUM+1;
    END;
  ELSE
    DO;
      /*? FUNC="DEL" */
      /*? カウントダウン */
      NUM=NUM-1;
    END;
  END;
  1012B
  1012C

```

【図14】

図14

図13に対応する概略PAD図



【図16】

図 16

